Dokploy GitHub Integration: Step-by-Step Guide (2026)

Modern application deployment has evolved significantly over the past few years. Instead of manually uploading files to servers, developers now rely on Git-based deployment workflows that automatically build and deploy applications whenever new code is pushed. This approach not only saves time but also reduces deployment errors and improves productivity.

Dokploy simplifies this process by offering seamless GitHub integration. With just a few configuration steps, you can connect your GitHub repository to Dokploy, automate deployments, and manage your applications directly from a user-friendly dashboard.

Whether you're deploying a personal project, SaaS application, or enterprise solution, integrating GitHub with Dokploy creates a streamlined Continuous Deployment (CD) workflow that keeps your applications up to date with minimal effort.

In this guide, you'll learn how to connect GitHub to Dokploy, configure automatic deployments, and troubleshoot common integration issues.

Why Integrate GitHub with Dokploy?

GitHub is one of the most widely used version control platforms, while Dokploy is a powerful self-hosted deployment platform. Combining the two allows developers to automate deployments and eliminate repetitive manual tasks.

Key benefits include:

- Automatic deployments after every code push
- Faster release cycles
- Improved collaboration
- Centralized deployment management
- Reduced deployment errors
- Easy rollback capabilities

This integration is especially useful for teams following modern DevOps and CI/CD practices.

Prerequisites

Before connecting GitHub to Dokploy, make sure you have:

- A running Dokploy installation
- A GitHub account
- A GitHub repository
- A configured server in Dokploy
- Docker installed on your server
- Administrator access to Dokploy

Once these requirements are met, you're ready to begin.

Step 1: Log in to Dokploy

Open your Dokploy dashboard in a web browser.

Log in using your administrator credentials.

After signing in, you'll see the main dashboard where applications, servers, and deployment settings are managed.

Step 2: Add Your Server

If you haven't already connected your VPS or dedicated server, navigate to:

Servers → Add Server

Provide:

- Server name
- IP address
- SSH credentials

Dokploy will verify the connection before making the server available for deployments.

Step 3: Connect Your GitHub Account

Navigate to the Git integration section inside Dokploy.

Choose **GitHub** as your Git provider.

Depending on your Dokploy version, you'll be asked to authenticate using:

- GitHub OAuth
- Personal Access Token (PAT)

Grant Dokploy permission to access the repositories you want to deploy.

For security, it's recommended to provide access only to the required repositories instead of your entire GitHub account.

Step 4: Select a Repository

Once GitHub is connected, Dokploy will display your available repositories.

Choose the project you want to deploy.

You'll also need to select:

- Repository branch (such as `main` or `master`)
- Deployment server
- Build configuration

This information tells Dokploy where to retrieve your application source code.

Step 5: Configure Build Settings

Each application may require different build instructions.

Common configuration options include:

- Build command
- Install command
- Start command
- Dockerfile location
- Environment variables

For example, a Node.js project may require:

- Install dependencies
- Build the project
- Start the production server

Dokploy automatically executes these steps during deployment.

Step 6: Add Environment Variables

Most applications require configuration values that should never be stored in source code.

Examples include:

- Database credentials
- API keys
- Authentication secrets
- Third-party service tokens

Navigate to the **Environment Variables** section and securely add all required values before deploying your application.

This helps maintain security while keeping deployment configurations organized.

Step 7: Deploy the Application

Once your repository and settings are configured, click **Deploy**.

Dokploy will automatically:

- Clone the repository
- Install dependencies
- Build the application
- Create Docker containers
- Launch the application

Deployment progress can be monitored directly from the dashboard.

Step 8: Enable Automatic Deployments

One of the biggest advantages of GitHub integration is deployment automation.

Enable automatic deployment or webhook support inside Dokploy.

After configuration:

1. Push new code to GitHub.
2. GitHub notifies Dokploy.
3. Dokploy starts a new deployment automatically.
4. The updated application becomes available.

This creates a smooth Continuous Deployment workflow without manual intervention.

Monitoring Deployment Status

Dokploy provides detailed deployment logs that allow developers to monitor every deployment.

You can review:

- Build logs
- Deployment history
- Container logs
- Error messages
- Deployment duration

These tools make debugging significantly easier if something goes wrong.

Common Integration Issues

Repository Not Visible

Ensure Dokploy has permission to access the repository through GitHub OAuth or your Personal Access Token.

Deployment Failed

Review build logs for:

- Missing dependencies
- Incorrect build commands

- Invalid Docker configuration
- Missing environment variables

Authentication Errors

Verify that your GitHub token hasn't expired and includes the necessary repository permissions.

Webhook Not Triggering

Check:

- Webhook configuration
- Repository permissions
- Server connectivity
- Firewall settings

Best Practices

To get the most out of GitHub integration:

- Use separate branches for development and production.
- Store secrets in environment variables instead of source code.
- Keep Docker images updated.
- Review deployment logs regularly.
- Enable automatic backups before major releases.
- Test deployments in a staging environment before production.

These practices improve both security and deployment reliability.

Benefits of GitHub Integration

Connecting GitHub with Dokploy offers several long-term advantages:

- Faster software releases
- Automated deployment workflows
- Improved collaboration
- Better version control
- Simplified rollback process
- Reduced manual configuration

For teams practicing Continuous Integration and Continuous Deployment (CI/CD), this integration can significantly improve productivity.

Conclusion

GitHub integration is one of Dokploy's most valuable features, allowing developers to automate deployments directly from their source code repositories. By connecting GitHub, configuring build settings, and enabling automatic deployments, you can eliminate repetitive manual tasks and create a reliable deployment pipeline.

Whether you're working on a personal project or managing production applications, GitHub integration helps ensure faster updates, consistent deployments, and a more efficient development workflow. If you're looking to simplify self-hosted application deployment, integrating GitHub with Dokploy is an excellent place to start.